TypeScript Meetup v0.2

10 min	Around the room introductions
10 min	Opening comments - Anders Hejlsberg
20 min	TypeScript in the trenches - Eric Gamma
20 min	TypeScript update - Luke Hoban
20 min	TypeScript Q&A session - Luke, Anders, Steve Lucco, and Team
20 min	**Break: Eat, drink, and Socialize**
10 min	Bing and TypeScript - Kiran Badam & Sarvesh Nagpal
10 min	Cloud Make: Using TypeScript for Builds and WorkFlows - Wolfram Schulte
10 min	TypeScript language experiments - Ron Buckton
10 min	Xbox Music and TypeScript - Dan Rodgers (not recorded/streamed)
10 min	Using TypeScript to simplify JavaScript hosting - Paul Vick (not recorded/streamed)

Contact: stevenic

Thank You!

TypeScript Momentum

Community

Over 3000 CodePlex posts, 100 forks, 1000 StackOverflow questions, and 300 feature requests

Ecosystem

Over 180 .d.ts library definitions covering more than 90% of popular JavaScript frameworks.

Tool Support

IDEs: WebStorm, WebMatrix, JSBin, Web Essentials, Sublime Text, vi, Emacs, FlashDevelop, Brackets
Build: Heroku, Ruby, grunt, ASP.NET, node, compile-in-client
Testing: Chutzpah, tsUnit
Preprocessing: Bakehouse
Typings management: tsd

TypeScript 0.9

Generics

Generic classes, interfaces, and methods: The top requested feature by users

Overloading on Constants

Examples include createElement, getElementsByTagName, addEventListener

Other Features

Enum types, 'export =', function/module and class/module merging, methods in object literals

Compiler

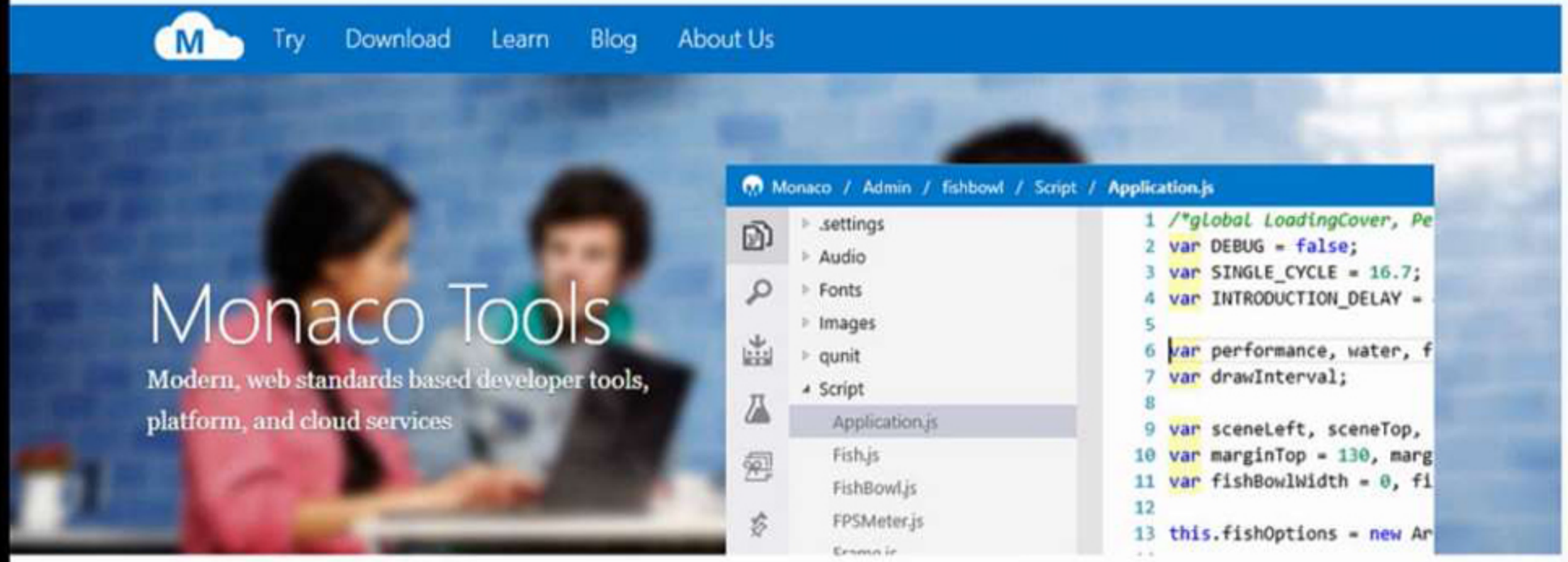
Incremental parser, pull-model type checker, scaling to 100K+ line projects

Application Scale JavaScript

TypeScript in the Trenches

Erich Gamma
Microsoft

http://monacotools



Easily get started

No long waits for installation and configuration. Just log in and get coding!

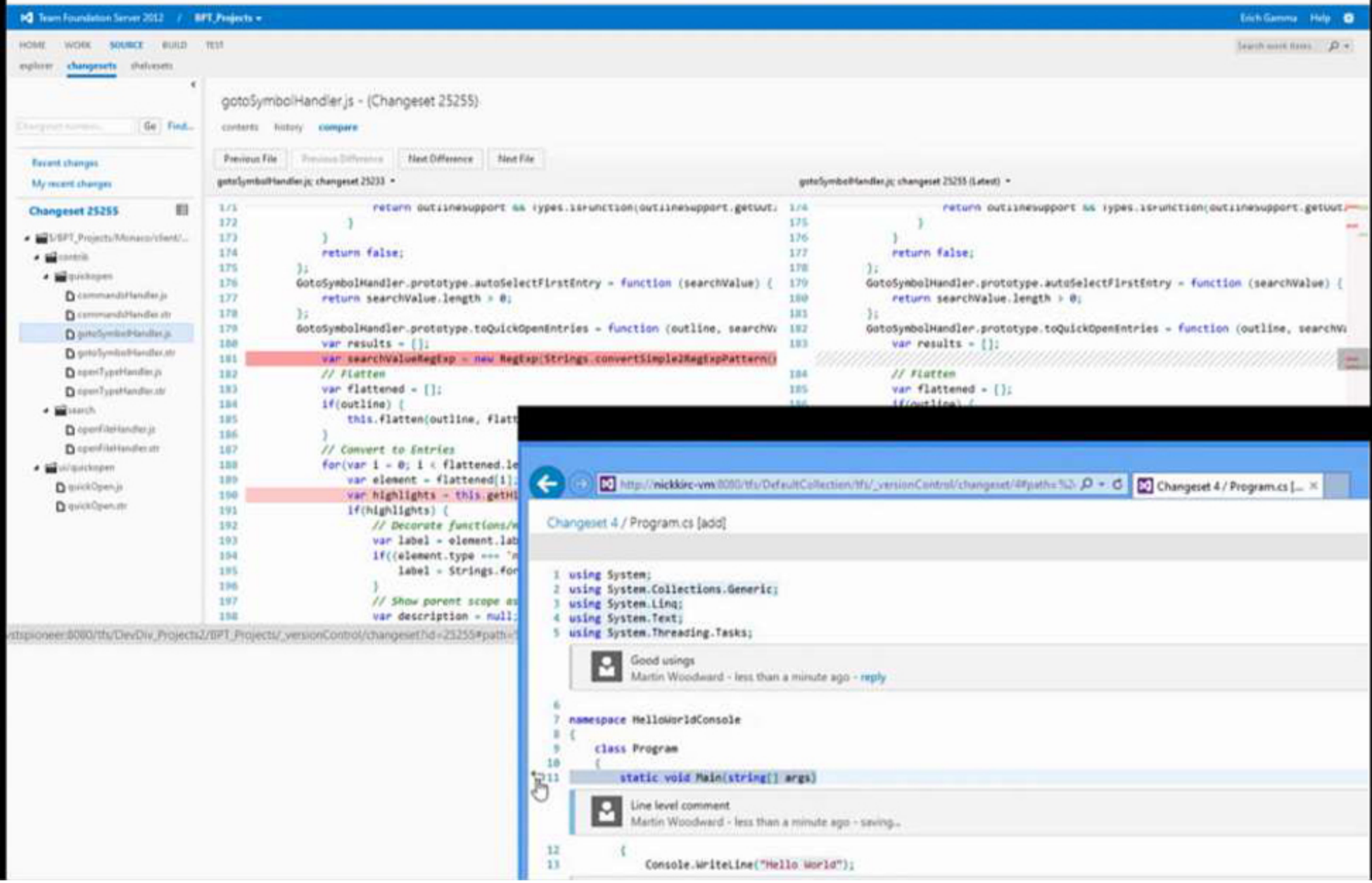
Code in the cloud with confidence

Develop online with rich, contextual features like IntelliSense, Source Assist, Quick Open and many more!

Platform of reusable components

We've built Monaco as a set of simple, yet extensible modules that you can adopt and tweak yourself, based on your requirements and scenarios

Team Foundation Server



dafny

Microsoft Research

Is this program correct? Ask dafny!

```
1  function Ackermann(m: int, n: int): int
2    // The following lexicographic pair allows Dafny to prove termination.
3    // Still, you may not want to sit around and wait for a call to Ackermann
4    // to terminate.
5    decreases m, n;
6  {
7    if m <= 0 then
8      n + 1
9    else if n <= 0 then
10     Ackermann(m - 1, 1)
11   else
12     Ackermann(m - 1, Ackermann(m, n - 1))
13 }
14
```

ask dafny

home | tutorial | video | permalink

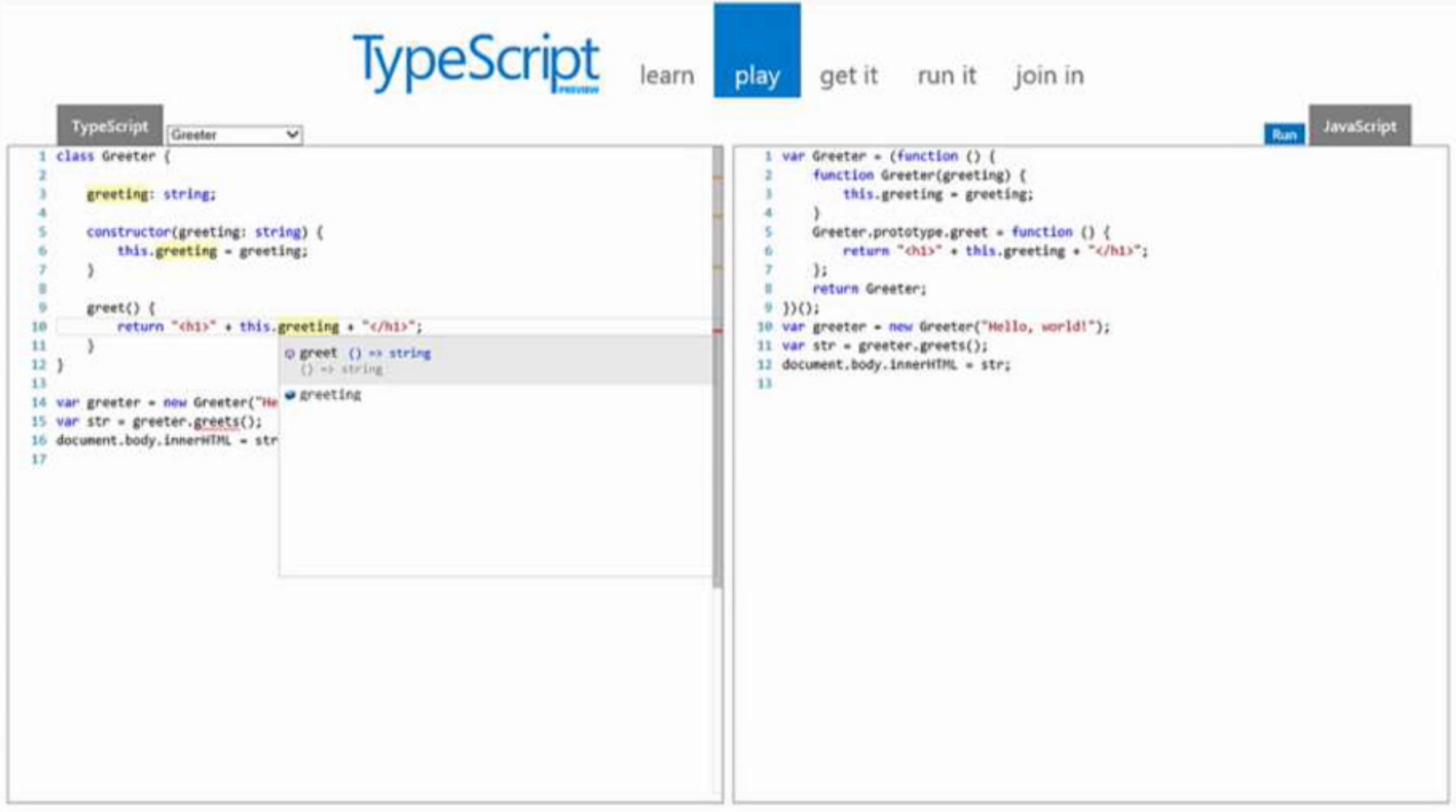
Dafny program verifier finished with 1 verified, 0 errors

samples
Hello
Fibonacci
CountToN

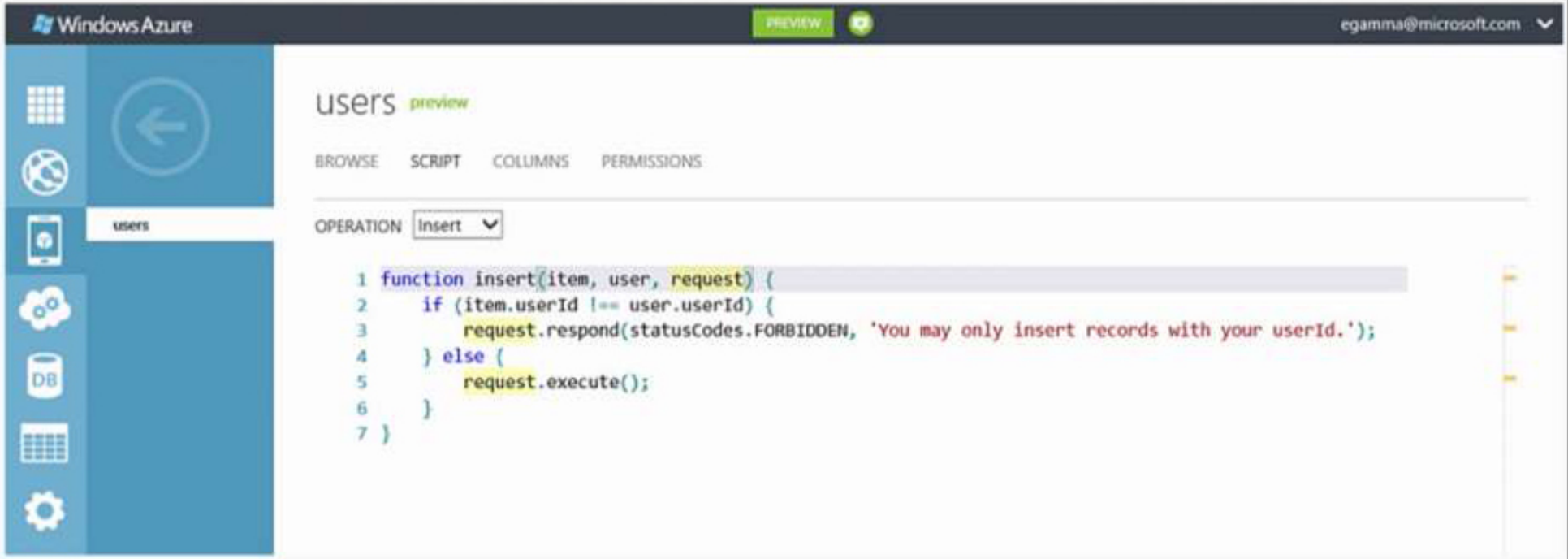
about Dafny - A language and program verifier for functional correctness

Dafny is an imperative, object-oriented programming language with classes and inductive datatypes, and specification constructs for describing intended behavior. The Dafny verifier checks that programs live up to their specifications.

TypeScript Playground



Azure


Windows Azure
PREVIEW
egamma@microsoft.com
users
users preview
BROWSE
SCRIPT
COLUMNS
PERMISSIONS
OPERATION
Insert
function insert(item, user, request) {
if (item.userId !== user.userId) {
request.respond(statusCodes.FORBIDDEN, 'You may only insert records with your userId.');
} else {
request.execute();
}
}

Office 365 - Napa

Office 365 ▸ Napa ▸ bubbles ▸ App.js — Erich Gamma

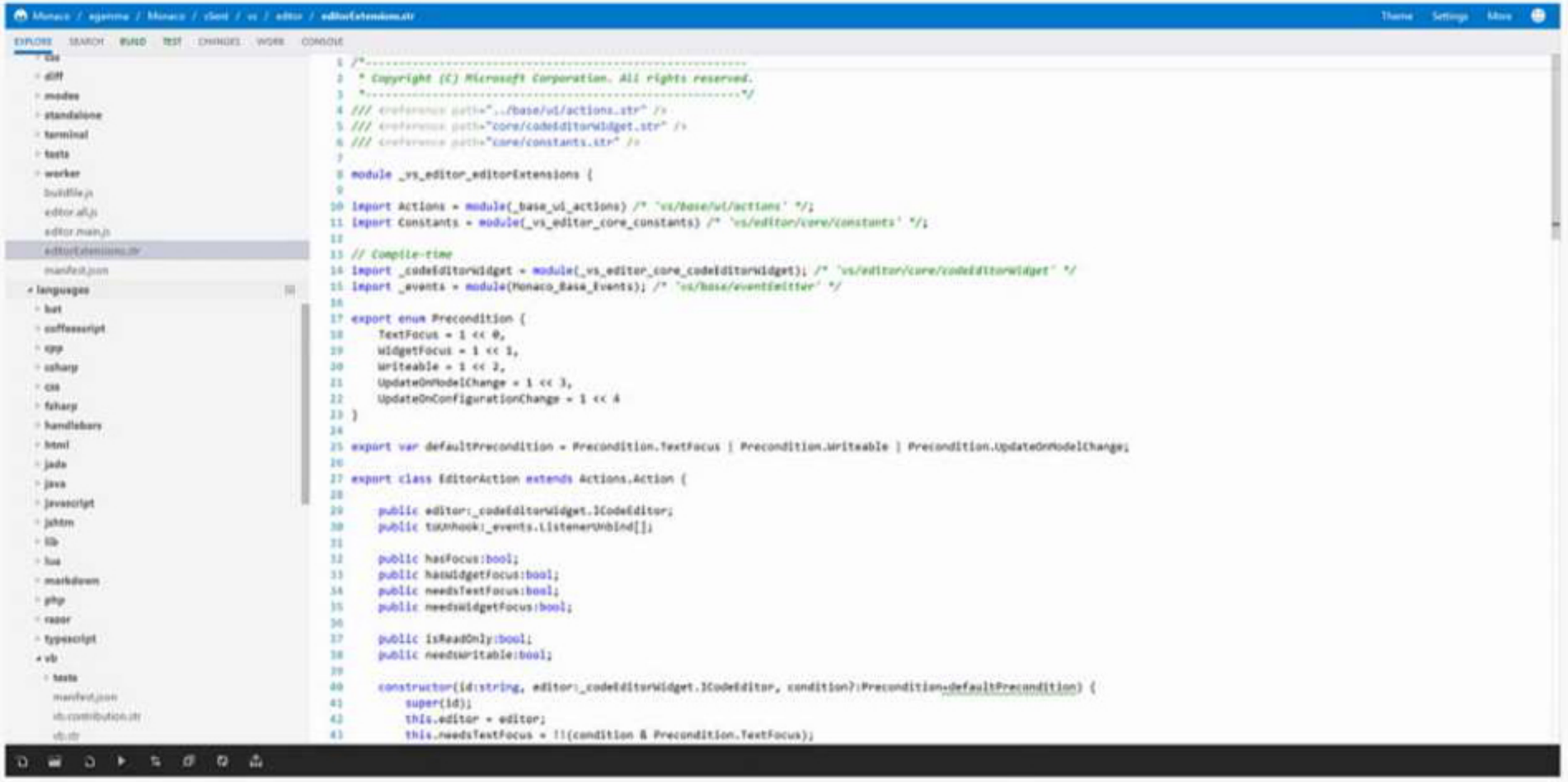
- Content
- Images
- OfficeDocuments
- Pages
- Scripts
  - App.js
  - bubbles.txt
  - processing-1.0.0.js

```
'use strict';
/*global Processing*/
var _values = [];
var cachedvalues = [['MSFT (cached)',27.34],['YHOO (cached)',19.85],['FB (cached)', 29.08]];

// The initialize function must be run each time a new page is loaded
Office.initialize = function(reason) {
    $(document).ready(function() {
        $('#read').click(getDataFromSelection);
        $('#write').click(writeDataToSelection);
    });
};

// Reads data from current document selection
function getDataFromSelection() {
    Office.context.document.getSelectedDataAsync(Office.CoercionType.Matrix, function(result) {
        Office.context.
        if (result.stat
            var yql =
            var queryUR
            $.getJSON(q
                var sto
                if (res
                    var
                    quo
                    $.e

                    });
                    _va
                } else
                    _values = cachedvalues;
                }
                //Display bubbles visualization
                showCanvas();
            });
        }
    });
```

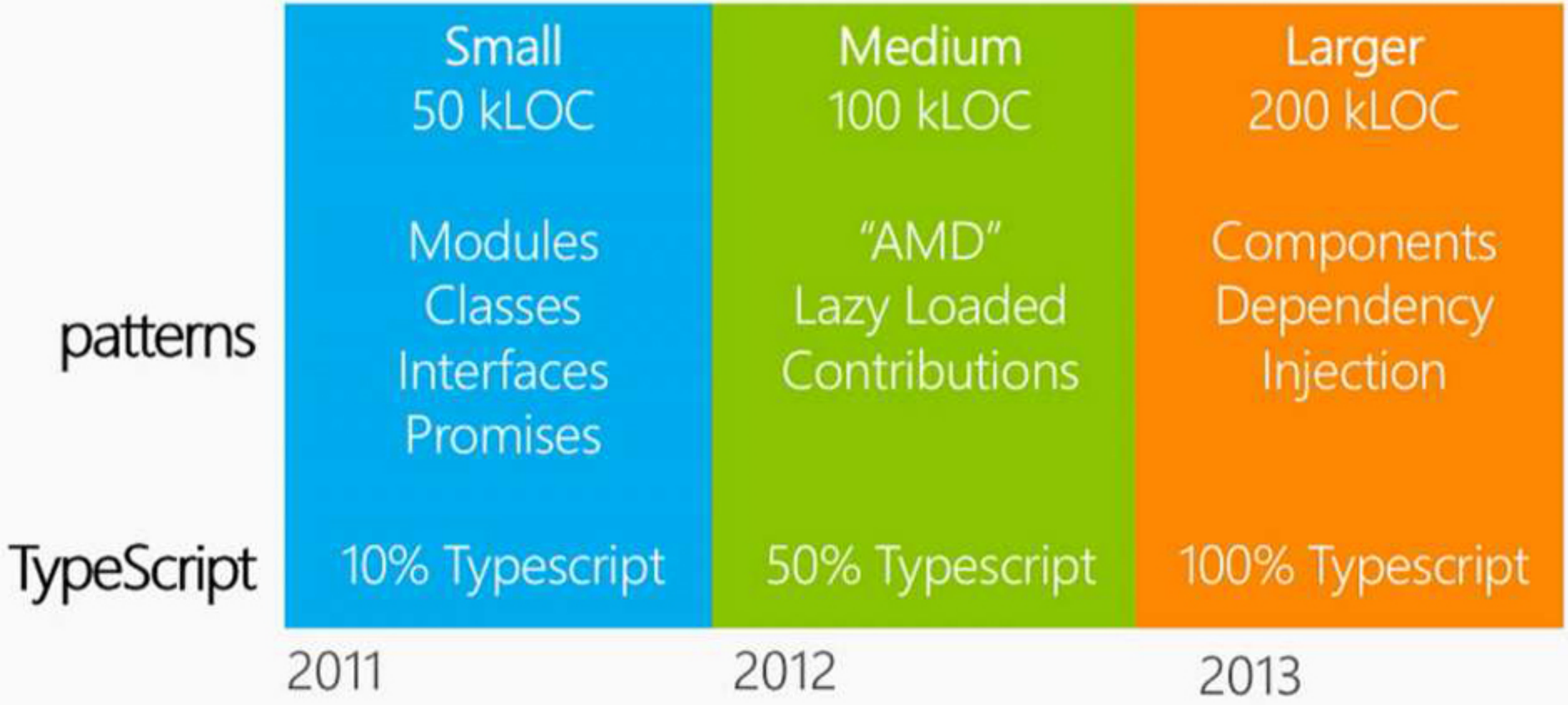
contentLanguage string
displayLanguage
document
license

result.value.join() + '\')';
+ '&format=json&env=http%3A%2F%2Fdatatables.org%2Fal

Monaco @ Work - Dog Fooding



Our Journey


Small
50 kLOC
Medium
100 kLOC
Larger
200 kLOC
patterns
Modules
Classes
Interfaces
Promises
"AMD"
Lazy Loaded
Contributions
Components
Dependency
Injection
TypeScript
10% Typescript
50% Typescript
100% Typescript
2011
2012
2013

Inside

Classes

```
interface IRenderer {
    getHeight(tree: ITree, element: any): number;
    render(tree: ITree, element: any): void;
}
```

Interfaces at Work

Options, JSON

```
interface IOptions {
    hideButtons?:bool;
    okOnFocusLost?:bool;
}
```

Interfaces at Work

Functions, Callbacks

```
interface IMeasurementCallback {
    (accessor: IMeasurementAccessor):void;
}
```

Interfaces at Work

Indexed access

```
interface IEventMap {
    [name: string]: EventListener[];
}
```

Interfaces at Work

External Types

```
interface JQuery {
  addClass(classNames: string): JQuery;
  addClass(func:(index: any, currentClass: any)=>JQuery);
  attr(attributeName: string): string;
  attr(attributeName: string, value: any): JQuery;
  ....
}
```

Demo

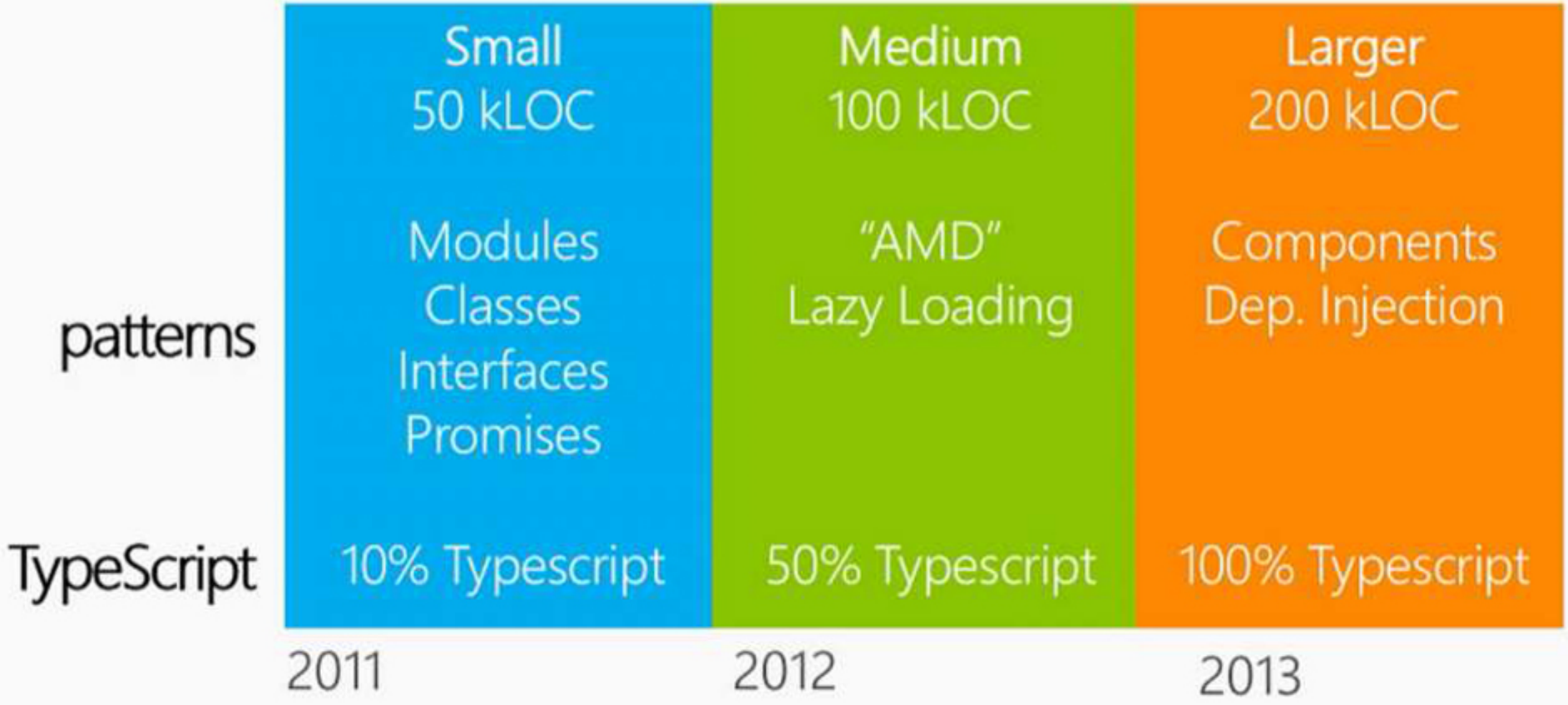
Type definitions

TypeScript Type Definitions

https://github.com/borisyankov/DefinitelyTyped

amcharts	2 months ago	amCharts have been added [covobonomo]
angularjs	4 days ago	angular.d.t.s ng.IModule use of Object instead of {}, added tests [DerAlbertCom]
async	2 months ago	fix async-tests errors [Diullei]
backbone	a month ago	Optional parameters for Backbone. [Guuz]
bgiframe	19 days ago	Definition of bgiframe for IE6 - simple plugin implementation without... [sumegizoltan]
bootbox	3 months ago	Conformed tests to readme.md [vbortone]
bootstrap-notify	2 months ago	Added type definitions for bootstrap-notify [niemyjski]
bootstrap.datepicker	3 months ago	Update reference paths [borisyankov]
bootstrap	13 hours ago	Added overload for modal so can use backdrop: "static" [shearnie]
box2d	a month ago	Updated box2d, underscore, sugar [jbaldwin]
breeze	9 days ago	breeze updated - 1.2.7 [Diullei]
casperjs	2 months ago	casperjs header was normalized. [Diullei]
chai	2 days ago	Included Chai Expect. [kazimanzurrashid]
cheerio	2 months ago	Update readme and metadata for Cheerio [borisyankov]
chosen	3 months ago	Update reference paths [borisyankov]
chrome	3 months ago	Add type information for Chrome's Socket API [ahf]
codemirror	3 months ago	Update reference paths [borisyankov]
commander	a month ago	Add CommanderJs definitions [mdezem]
d3	11 days ago	Add more D3 Transition methods [rblrkby]
domo	2 months ago	Update domo definitions readme and metadata [borisyankov]

Our Journey


Small
50 kLOC
Medium
100 kLOC
Larger
200 kLOC
patterns
Modules
Classes
Interfaces
Promises
"AMD"
Lazy Loading
Components
Dep. Injection
TypeScript
10% Typescript
50% Typescript
100% Typescript
2011
2012
2013

Growing Pains

Managing scripts and their order

Manually edited lists of scripts

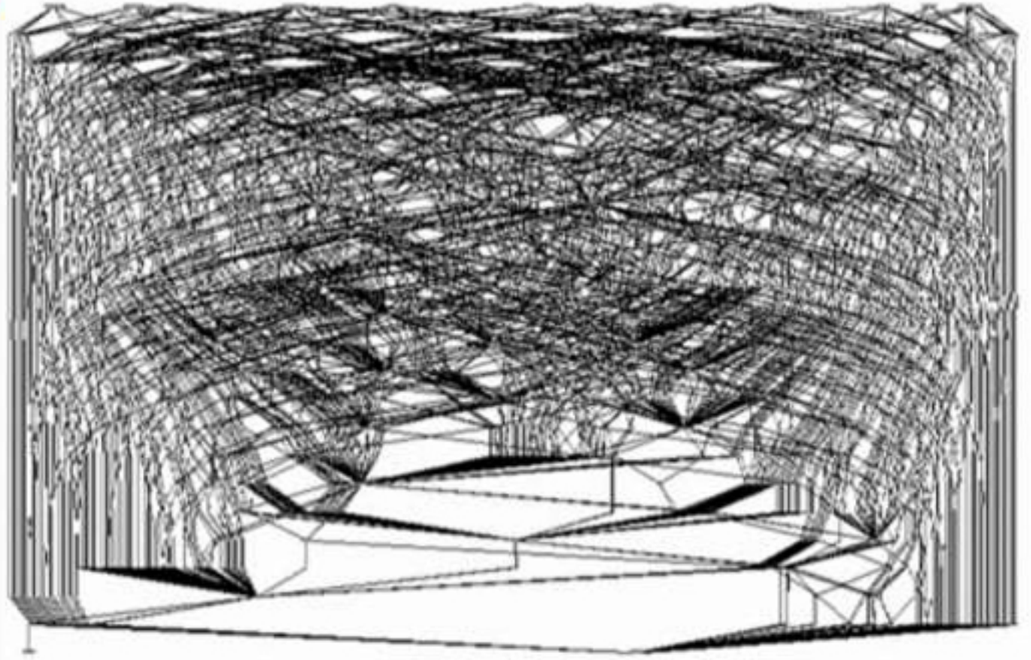
/ide/**manifest.json**

```
27
28      "src": {
29          "/monaco/ide/public/js/workbench.js": [
30              "public/js/constants.js",
31              "public/js/platform.js",
32              "public/js/events.js",
33              "public/js/core/context.js",
34              "public/js/core/storage.js",
35              "public/js/core/history.js",
36              "public/js/model/workspaceModel.js",
37              "public/js/ui/sitelet.js",
38              "public/js/ui/commands.js",
39              "public/js/ui/memento.js",
40              "public/js/ui/layout.js",
41              "public/js/ui/actions/action.js",
42              "public/js/ui/menu.js",
43              "public/js/ui/viewlets/viewlet.js",
44              "public/js/ui/parts/part.js",
45              "public/js/ui/parts/sidebarPart.js",
46              "public/js/ui/parts/editor/baseEditor.js",
47              "public/js/ui/parts/editor/editorModel.js",
48              "public/js/ui/parts/editor/editorInput.js",
49              "public/js/ui/parts/editor/editorOptions.js",
50              "public/js/ui/parts/editor/textEditor.js",
51              "public/js/ui/parts/editor/stringEditorModel.js",
52              "public/js/ui/parts/editor/stringEditorInput.js",
```

Growing Pains: Dependencies…

It is easy to import an internal module…

It is easy to contribute to a module…



"our dependency graph was such a mess that each area had a dependency on just about every other area." –Nick

AMD to the Rescue

define(id?, dependencies?, factory)

```
define([
    'vs/base/lib/winjs', 'vs/editor/zoneWidget'],
    function(WinJS, ZoneWidget) { ... }
);
```

TypeScript: External Modules

TypeScript supports code generation for different module systems

```
import widget= import('vs/base/widget');

import http= import('http');
```

```
tsc --module amd app.ts
```

```
define([…], function(…) {…}
```

```
tsc --module commonjs HttpServer.ts
```

```
var http= require("http");
```

Before/After

AMD in JavaScript

```
define(['..../winjs.base', '..../zoneWidget'],
    function(WinJS, ZoneWidget) { ... }
);
```

TypeScript

```
import WinJS= import('vs/base/lib/winjs');
import ZoneWidget = import('vs/editor/zoneWidget');
```

AMD Applied – Self Contained Modules

Support à la carte consumption

Expressing CSS dependencies

CSS AMD loader plugin
TypeScript pragma

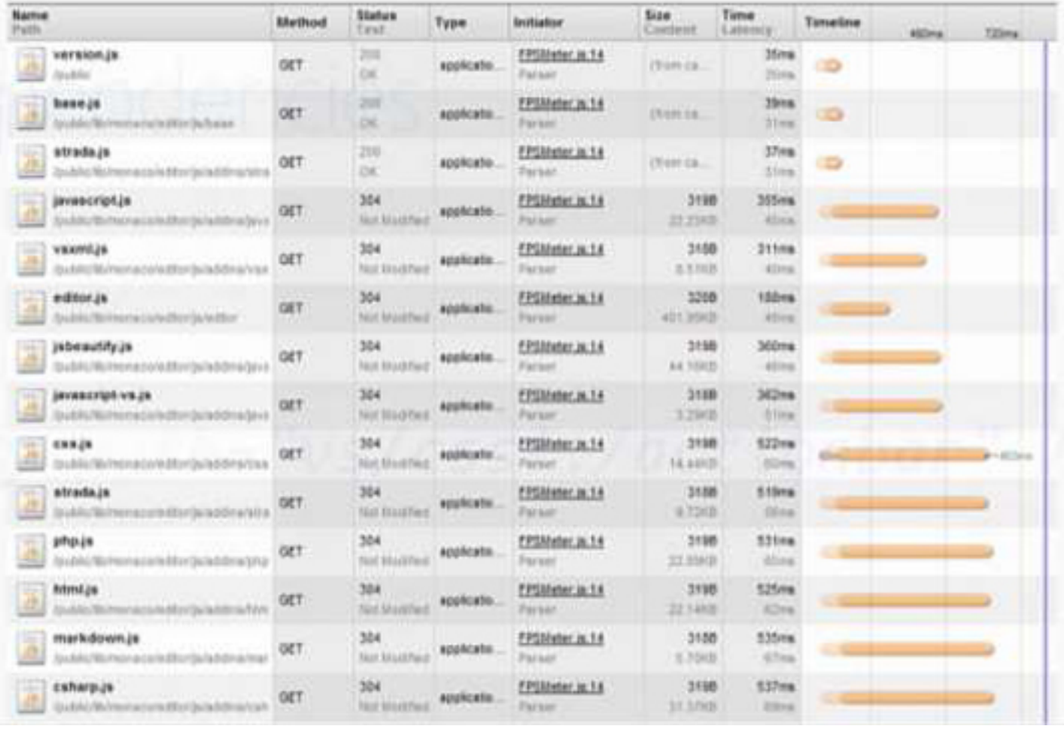
```
/// <amd-dependency path="vs/css!./actionbar" />
```

Growing Pains

Start-up time

Eager loading of scripts and CSS

Slow startup: #requests, data transfer



Lazy Loading Contributions

csharp.contribution.ts

```
modeRegistry.registerMode(
    ['text/x-csharp'],
    new Platform.Descriptor(
        'vs/languages/csharp/csharp',
        'CSMode')
);
```

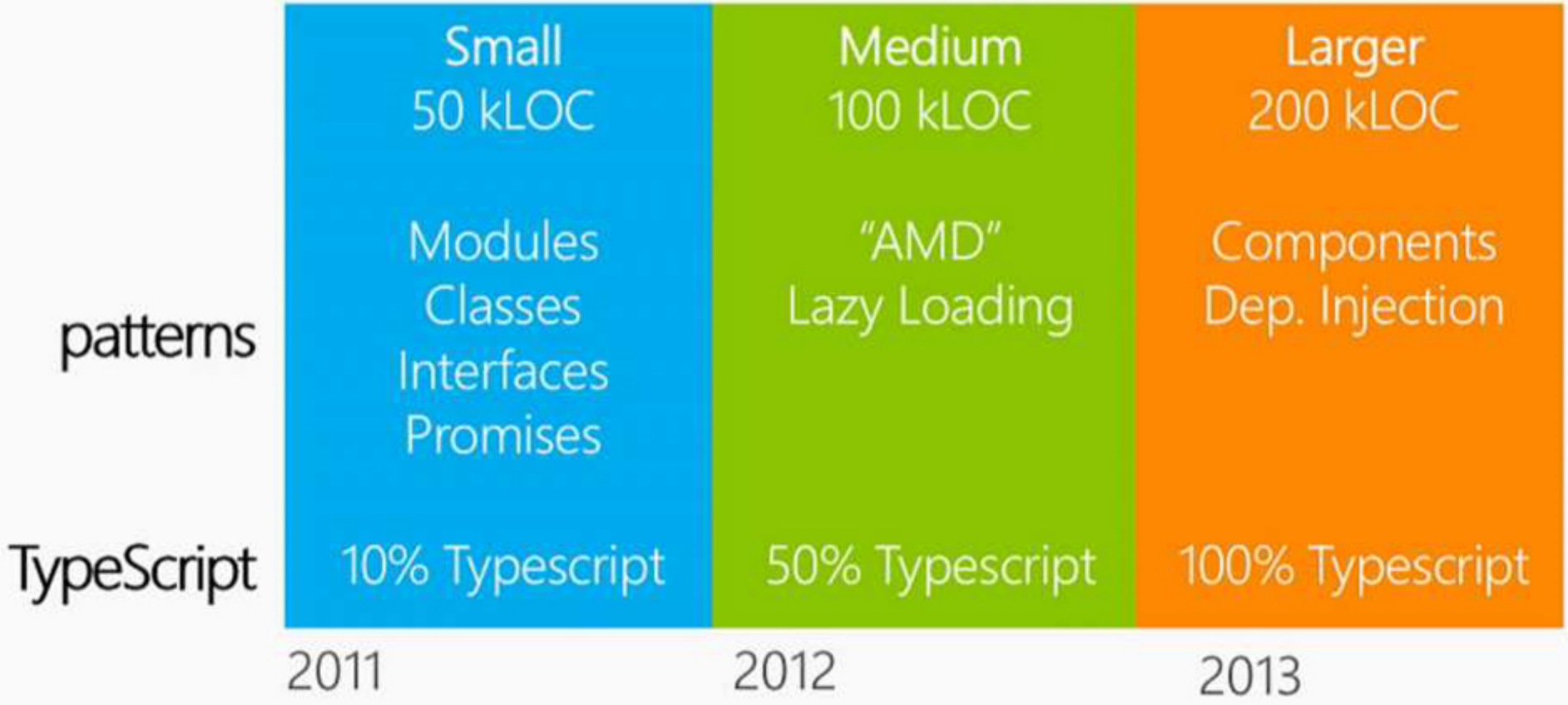
csharp.ts

```
export class CSMode extends
modesExtensions.AbstractMode {
    constructor() {
        super('vs.languages.csharp');
    }
   // lots of code ....
}
```

After the AMD Migration Impressions

"It feels like fresh showered. Self contained modules, no more cycles, no more globals, clean file system structure" -Alex

Our Journey


Small
50 kLOC
Medium
100 kLOC
Larger
200 kLOC
patterns
Modules
Classes
Interfaces
Promises
"AMD"
Lazy Loading
Components
Dep. Injection
TypeScript
10% Typescript
50% Typescript
100% Typescript
2011
2012
2013

100% TypeScript

"Writing JavaScript code in a large project is like carving code in stone" –Alex

Migration is code clean-up and real work

Velocity around 300 LOCs per hour

"As I did conversions, I began typing various object literals I was passing around as interfaces. Soon enough, I realized how inconsistent I was, the same data was flowing around in at least 3 different formats. This is because of the easiness through which you can create literals in JavaScript .... Need some placeholder for data?... Just create a new literal object." --Alex

Even More Growing Pains

"Writing JavaScript code in a large project is like carving code in stone" –Alex

Tooling slow down, translation time

Tools need to digest lots of source to infer types

Hard coded dependencies on context

Decreased testability

Difficulty to reuse components in different contexts

Componentization

Tooling slow down, translation time

Tools need to digest lots of source to infer types

Reuse TypeScript code as 'binary' JS components with a declarations file

Example using TypeScript language services as a component

Compiler and language services > 30kLOC of TypeScript

```
tsc –declarations –out typescriptservices.js typescript.ts
```

Dependency Injection

Dependencies on *services* provided by container

Decreased testability

Need for configuration flexibility

Services (20)

Selection, Progress, Logging, History, Events, Configuration, Telemetry, Requests, …

Dependency Injection

```
export interface IProgressService {
    show(total?:number, delay?:number):IProgressRunner;
}

export interface IProgressServiceConsumer {
    injectProgressService(service:IProgressService):void;
}

export class Explorer implements Services.IProgressServiceConsumer{
    private progressService:Services.IProgressService;
    public injectProgressService(service:Services.IProgressService) {
        this.progressService = service;
    }
    //....
}
```

TS Retrospective

We were on the bleeding edge...

...but we expected it and had plenty of band aid

We would do it again, the benefits outweigh the pains

Code readability, refactoring agility, tooling, fun

We would start with TypeScript (and AMD) from the beginning

http://www.typescriptlang.org/